814426_1
DJT/BJS/cb
August 27, 2008

PATENT APPLICATION
Docket No.: 3336.1016-002

Applicants: Foster D. Hinshaw, David L. Meyers, and Barry M. Zane

Application No.: 10/668,113 Group: 2162

Filed: September 18, 2003 Examiner: Jean B. Fleurantin

Confirmation No.: 7658

For: PROGRAMMABLE STREAMING DATA PROCESSOR FOR DATABASE APPLIANCE HAVING MULTIPLE PROCESSING UNIT GROUPS

OIPE
AUG 2 9 2008
PATENT & TRADEMARK OFFICE

CERTIFICATE OF MAILING OR TRANSMISSION

I hereby certify that this correspondence is being deposited with the United States Postal Service with sufficient postage as First Class Mail in an envelope addressed to Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, or is being facsimile transmitted to the United States Patent and Trademark Office on:

AUGUST 27, 2008
Date

Signature

CHRISTINE A. BUDD
Typed or printed name of person signing certificate

APPEAL BRIEF

Mail Stop Appeal Brief Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

This Appeal Brief is submitted pursuant to the Notice of Appeal received in the U.S. Patent and Trademark Office on May 27, 2008, and in support of the appeal from the final rejection set forth in the Office Action mailed on January 23, 2008. The fee for filing a brief in support of an appeal is enclosed. A Petition for Extension of Time and the appropriate fee are being filed concurrently.

08/29/2008 HLE333 00000063 10668113

01 FC:1402 510.00 OP
02 FC:1251 120.00 OP

TABLE OF CONTENTS

I. REAL PARTY IN INTEREST

The real party in interest is Netezza Corporation, 200 Crossing Boulevard, Framingham, Massachusetts 01702-4480. Netezza Corporation is the Assignee of the entire right, title and interest in the subject application, by virtue of an Assignment recorded on March 5, 2004 at Reel 015039, Frames 0125-0128.

II. RELATED APPEALS AND INTERFERENCES

Appellants, the undersigned Attorney and Assignee are not aware of any related appeals or interferences which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

III. STATUS OF CLAIMS

Claims 1 through 31 have been finally rejected, and a copy appears in the Appendix of this Brief. Claims 1, 3, 6, 10, 16, 17, 20, 21, 23 and 30 were amended in the Amendment filed on November 9, 2007. Claims 2, 4, 5, 7-9, 11-15, 18, 19, 22, 24-29 and 31 appear as originally filed. Claims 1-31 are being appealed herein.

IV. STATUS OF AMENDMENTS

No amendments are being filed concurrently with this appeal brief.

V. SUMMARY OF CLAIMED SUBJECT MATTER

<u>Claim 1</u>

The invention according to Claim 1 is an asymmetric data processor comprising at least a first group of nodes, a second group of nodes, and a network connecting the nodes. The first group of nodes includes one or more host computers. The one or more host computers accept queries for data, and transform those queries into one or more jobs.

The second group of nodes comprises a plurality of Job Processing Units (JPUs). Each JPU comprises a memory, a network interface, a streaming data interface, one or more general-purpose CPUs, and one or more Programmable Streaming Data Processors (PSDPs). The CPUs are configured to respond to requests from at least one of the host computers, as well as from

other JPUs in the second group of nodes. The PSDPs are configured to perform filtering functions directly on data received from the streaming data interface, thereby performing initial processing of data. Each JPU is configured to receive jobs from one or more nodes in the first group, perform work requested by the jobs, and generate a result based on the work. (See at least Specification at page 12, line 5 – page 16, line 28 with reference to Fig. 1 and components 10, 12, 20, 22, 26, 27 and 28; and page 28, lines 7 – page 30, line 22 with reference to Fig. 4 and components 20, 28, 281, 282 and 301.)

Claim 9

Claim 9 depends from Claim 1, described above, and further provides that the PSDP output data may contain projected fields not contained in the source data. The projected fields may include, for example, a row address, transforms, results of expression evaluation, results of bit joins, and results of visibility tests. (See at least Specification at page 28, line 27 – page 29, line 22; page 36, lines 2-7; and page 39, lines 21-29.)

Claim 12

Claim 12 depends from Claim 1, described above, and further provides that the PDSP performs a join operation. The field values being joined have a small range of values, so that the presence or absence of a particular value can then be encoded as a bit within a sequence of bits, whose position within the sequence corresponds to the field value. (See at least Specification at page 9, lines 15-22.)

Claim 13

Claim 13 depends from Claim 1, described above, and further provides that the PDSP performs an "exist join" operation. The field values being joined in an "exist join" operation may have a small range of values, so that the presence or absence of a particular value can then be encoded as a bit within a sequence of bits, whose position within the sequence corresponds to the field value. (See at least Specification at page 9, lines 15-22 and page 39, lines 4-13.)

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

A. Whether Claims 1-8, 10, 11 and 14-31 are properly rejected under 35 U.S.C. § 102(e) as being anticipated by Kabra et al. (U.S. Patent No. 6,507,834).

B. Whether Claims 9, 12 and 13 are properly rejected under 35 U.S.C. § 103(a) as being unpatentable over Kabra in view of Ozbutun (U.S. Patent No. 6,658,405).

VII. ARGUMENT

A. Rejection of Claims 1-8, 10, 11 and 14-31 Under 35 U.S.C. § 102(e)

Claims 1-8, 10, 11 and 14-31 have been rejected under 35 U.S.C. § 102(e) as being anticipated by Kabra.

Kabra discloses a method for parallel execution of queries at multiple data servers (Kabra, Abstract). As shown in Fig. 1, a client process 102, such as a computer application, issues a query for data, the query being formatted in a Structured Query Language (SQL). A query coordinator (QC) 104 receives the query from a client process 102 (See *Kabra* at col. 7, lines 27-37). The QC 104 generates an execution plan for the query and transmits portions of the plan to several data servers 130A-E. Each of the data servers 130A-E receives a portion of the plan corresponding to data stored at respective data storage devices 132A-E (*Id.* at col. 7, lines 4-12). The QC 104 controls the parallel execution of the plan on the data servers 130A-E, and collects results of the execution for delivery to the client process 102 (*Id.* at col. 7, lines 27-37).

Fig. 6A of Kabra illustrates the above in further detail, where the query coordinator 104 receives a message from the client 102 and, based on this message, generates an execution plan 608 (*Id.* at col. 11, 50-61). The data servers 130A-B each receive a portion of this execution plan 610A-B, and execute the respective portion of the plan 612A-B to produce query results (*Id.* at col. 11, lines 26-39). The data servers 130A-B transmit the results to the QC 104, which compiles the results and transmits the compiled query results 614 to the client 102.

Referring to Fig. 2 of Kabra, the query coordinator 104 includes a query scheduler 122, which in turn includes a parallelizer 202 and dispatcher 204. The parallelizer 202 receives a plan and generates a parallel execution plan having a number of segments (*Id.* at Fig. 3 and col. 9, lines 9-20). Each segment of the parallel execution plan can be executed concurrently by different data servers 130A-E (*Id.*). As a result, a plan to respond to a query can be completed in parallel. Kabra's parallelism is illustrated in Fig. 6A, described above, where each of the data

servers 130A-B receives (610A-B) and executes (612A-B) a different portion of the plan in parallel.

In order for a claim to be considered unpatentable under 35 U.S.C. §102(e), the reference must exactly teach each and every element of the claim. As stated by the Federal Circuit, "(a)nticipation under 35 U.S.C. § 102 requires the disclosure in a single piece of prior art of each and every limitation of a claimed invention." *Apple Computer Inc. v. Articulate Sys. Inc.*, 234 F.3d 14, 20, 57 USPQ2d 1057, 1061 (Fed. Cir. 2000). Kabra fails to teach every element of Applicant's Claims 1-8, 10, 11 and 14-31 at least for the reasons indicated below.

1. Kabra fails to teach a JPU having one or more CPUs that is responsive to requests from other JPUs.

Kabra fails to teach or suggest the feature of a JPU comprising "one or more general purpose CPUs, for responding to requests from at least one host computer in the first group, and to requests from other JPUs in the second group," as recited in Claim 1.

To aid in understanding the present invention, Applicants refer to an exemplary embodiment in Figs. 1 and 7 of their Specification. Here, a first group of nodes 10 includes a host computer 12, and a second group of nodes 20 includes a number of Job Processing Units (JPUs) 22. The host computer 12 communicates with Job Processing Units (JPUs) 22, where each JPU 22 may access data at respective storage devices 23 (*Specification* at page 12, lines 5-21 and page 13, lines 24-29).

In an example operation, the host computer 12 receives queries from a requester 33, 36, 38 (e.g., a client computer or application) to process data stored at a plurality of storage devices 23 (e.g. hard disk drives). A plan generator (204 at Fig. 3) at the host computer 12 generates a plan for processing the request (*Id.* at page 18, lines 14-19 and page 45, lines 1-5). The plan comprises a number of jobs, which are distributed among the JPUs 22-1 – 22-3 (Fig. 7) and host computer 12 (*Id.* at page 46, lines 5-8). Each job further comprises a sequence of instructions that are executed by the JPUs 22-1 – 22-3.

An example plan comprising a number of jobs is shown in the Specification at page 45, lines 6-36, where each job 1-7 includes a number of operations to be completed by a JPU 22 or host 12. In completing these jobs, each JPU 22 accepts and responds to requests from the host

12 and from other JPUs 22 (Specification, page 7, lines 1-6 and page 14, lines 7-27). For example, a first JPU 22-1 may complete Job 2, wherein it retrieves a data set and broadcasts a restricted data set to other JPUs as "TEMPStore" (*Id.* at page 46, lines 11-14). A second JPU 22-2 may complete Job 4, wherein it receives the "TEMPStore" data set from the first JPU 22-1 and projects a number of data fields. Thus, the JPUs 22 are responsive "to requests from at least one host computer in the first group, and to requests from other JPUs in the second group," as recited in Claim 1.

Kabra does not disclose an asymmetric data processor as recited in Claim 1. As stated above, Kabra describes a method of coordinating parallel execution of a query on multiple data servers. In contrast to the JPU of the present invention, however, the data servers 130 of Kabra are not responsive to "requests from <u>other JPUs</u>" to process data (emphasis added). As shown in Fig. 6A, each data server 130A-B executes a respective portion of the plan without communicating with one another. The Final Office Action, mailed January 23, 2008, at page 6, cites Kabra at col. 11, lines 50-52 and Fig. 6A for disclosing CPUs that are responsive "to requests from other JPUs in the second group." However, this passage of Kabra merely describes a request by a <u>client</u> for data, and does not disclose communication between data servers 130A-B.

On the contrary, Kabra teaches away from such communication: "...it is difficult to obtain and execute queries from within...one data server 130 when the data that is the subject of the [query] may reside on different data servers" (*Kabra* at col. 12, line 65 – col. 13, line 3). Kabra teaches "preprocessing" at the QC 104 to avoid a need to access data on one data server from another data server 130 (col. 13, lines 14-24). In contrast, embodiments of the present invention do not encounter such a problem because each JPU is responsive to "requests from other JPUs" to process data. Thus, Kabra actually teaches away from the present invention, and fails to disclose a JPU having a CPU "for responding to requests...from other JPUs in the second group" as recited in Claim 1.

Claims 2-8, 10, 11 and 14-31 depend from Claim 1 and thus inherit the limitations described above. Therefore, Applicants respectfully submit that the Examiner has failed to establish a *prima facie* case for rejection of Claims 1-8, 10, 11 and 14-31 under 35 U.S.C. § 102(e).

2. Kabra fails to teach a JPU having one or more Programmable Streaming Data Processors (PSDPs) that is responsive to requests from other JPUs.

Kabra fails to teach or suggest the feature of a JPU comprising "one or more Programmable Streaming Data Processors (PSDPs)-configured to perform filtering functions directly on data received from the streaming data interface, each PSDP thus performing initial processing on a set of data," as recited in Claim 1.

Referring again to Applicant's Specification at Fig. 1, each JPU 22 includes a Programmable Streaming Data Processor (PDSP) 28. The PDSP 28 serves as an interface between the JPU 22 and the storage device 23, and can filter data retrieved from the storage device 23 (*Specification* at page 7, lines 19-26). Such filtering operations are also performed in a streaming fashion, "on the fly" as the data is read as records streaming from the storage device 23 (*Id.* at page 8, lines 26-28). Once completed, results of the data processing may then be transmitted to the requester 33, 36, 38.

An example PSDP operation is illustrated in Applicant's Fig. 7. Here, each JPU 22 retrieves record data from respective registers 702. Each PSDP 28 performs filtering on the received record data, performing a "RESTRICT" operation such that only records meeting given criteria may pass to each JPU 22 (*Id.* at page 46, lines 22-29). As a result, the first and third PSDPs 28-1, 28-3 filter out all records from respective registers 702, while the second PSDP 28-2 passes all records from its register 702 to the corresponding JPU 22-2.

Kabra does not disclose a Programmable Streaming Data Processor (PSDP). The The Final Office Action, mailed January 23, 2008, refers to a "direct data transfer module" and a "valise" in Kabra as referring to a PSDP (*Kabra* at col. 10, lines 49-50 and col. 9, lines 31-34). Yet neither component of Kabra relates to a PSDP. A PSDP, as defined in Applicant's Specification, is "programmable to also interpret data in a specific format," through which "data can be filtered… in a streaming fashion, as data is read as records stream out of" a connected input (*Specification* at page 7, lines 19-26 and page 8, lines 26-28). No component in Kabra is disclosed as having such programmable and streaming operation. Thus, Kabra fails to disclose a PSDP as recited in Claim 1.

Claims 2-8, 10, 11 and 14-31 each depend from Claim 1 and thus are allowable at least for the reasons stated above. Therefore, Applicants respectfully submit that the Examiner has

failed to establish a *prima facie* case for rejection of Claims 1-8, 10, 11 and 14-31 under 35 U.S.C. § 102(e).

B. Rejection of Claims 9, 12 and 13 Under 35 U.S.C. § 103(a)

Claims 9, 12 and 13 have been rejected under 35 U.S.C. § 103(a) have been rejected as unpatentable over Kabra in view of Ozbutun (U.S. Patent No. 6,658,405).

Kabra, as described above, discloses a method for parallel execution of queries at multiple data servers. Ozbutun discloses methods for indexing bodies of records (*Ozbutun*, Abstract), and does not relate to a data processor having first and second groups of nodes; nor does it relate to a programmable streaming data processor.

Claims 9, 12 and 13, described above, each depend from Claims 1 and 2 and are directed to features of a programmable streaming data processor (PSDP). Neither Ozbutun nor Kabra teach or suggest a PSDP as recited in these claims.

Claim 9 depends from Claims 1 and 2 and adds that the "PSDP output data may contain projected fields not contained in the source data." Ozbutun merely describes indexing records using a bitmap index, where entries in the bitmap associate a range with a bitmap (*Ozbutun*, Abstract and col. 4, lines 50-64). Because Ozbutun fails to disclose a PSDP, he cannot describe PSDP output data.

Claims 12 and 13 depend from Claims 1 and 2 and add that a selected PSDP may perform a join operation or an "exist join" operation, respectively. Ozbutun describes combining bitmap indexes (*Ozbutun*, col. 4, lines 54-64), and fails to describe a selected PSDP performing a join or "exist join" operation as recited in Claims 12 and 13.

Given the aforementioned shortcomings of Kabra and Ozbutun, no combination of Kabra and Ozbutun teaches or suggests the present invention as recited in Claims 9, 12 and 13. Applicants, therefore, respectfully submit that the Examiner has failed to make out a prima facie case for the rejection of Claims 9, 12 and 13 under 35 U.S.C. § 103(a).

In view of the foregoing arguments, Applicants respectfully submit that all claims remaining in the application are in condition for allowance. Reversal of the rejections is requested so the application may pass to issue.

Respectfully submitted,

HAMILTON, BROOK, SMITH & REYNOLDS, P.C.

By

Benjamin J. Sparrow
Registration No.: 62,259
Telephone: (978) 341-0036
Facsimile: (978) 341-0136

Concord, MA 01742-9133
Dated: 8/27/08

CLAIMS APPENDIX

1. An asymmetric data processor comprising:

a first group of nodes comprising one or more host processors, each host comprising a memory, a network interface, and one or more Central Processing Units (CPUs), wherein each host accepts and responds to queries for data, and transforms such queries into one or more jobs;

a second group of nodes comprising a plurality of Job Processing Units (JPUs), wherein each JPU comprises:

a memory, for storing data;

a network interface, for receiving data and instructions;

a streaming data interface, for receiving data from a streaming data source;

one or more general purpose CPUs, for responding to requests from at least one host computer in the first group, and to requests from other JPUs in the second group; and

one or more Programmable Streaming Data Processors (PSDPs) configured to perform filtering functions directly on data received from the streaming data interface, each PSDP thus performing initial processing on a set of data; and

a network connecting the nodes within each group and between the two groups; and

wherein a JPU at the second group of nodes is configured to receive jobs from one or more nodes in the first group, perform work requested by the jobs, and generate a result based on the work.

2. The apparatus of claim 1 wherein the data comprises structured records, and the structured records further comprise fields of various lengths and data types.

3. An apparatus as in claim 1 wherein the filtering functions performed by the PSDPs comprise field-level filtering.

4. An apparatus as in claim 1 wherein the streaming data interface is an industry standard mass storage interface.

5. An apparatus as in claim 2 in which at least one selected PSDP performs Boolean comparisons of record field values against other values.

6. An apparatus as in claim 5 wherein the Boolean comparison is against at least one of other record field values and values held internally to that PSDP.

7. An apparatus as in claim 5 in which the selected PSDP restricts records that fail Boolean comparisons of field values, as such records stream into the PSDP and before such records are placed into the memory of the associated JPU.

8. An apparatus as in claim 2 in which the selected PSDP filters out fields of records that are not needed for particular queries, as such fields stream into the PSDP and before such fields are placed into the memory of the associated JPU, projecting forward into JPU memory those fields that are needed.

9. An apparatus as in claim 2 in which the PSDP output data may contain projected fields not contained in the source data, such as row address, transforms, results of expression evaluation, results of bit joins, and results of visibility tests.

10. An apparatus as in claim 2 in which a selected PSDP decompresses at least one of fields and records.

11. An apparatus as in claim 1 wherein the streaming data interface is connected to receive data from a peripheral device selected from the group consisting of disk drive, network interface, and other streaming data source.

12. An apparatus as in claim 2 in which a selected PSDP performs a join operation, where the field values being joined have a small range of values, so that the presence or absence of a particular value can then be encoded as a bit within a sequence of bits, whose position within the sequence corresponds to the field value.

13. An apparatus as in claim 2 in which a selected PSDP performs an "exist join" operation, where the field values being joined have a small range of values, so that the presence or absence of a particular value can then be encoded as a bit within a sequence of bits, whose position within the sequence corresponds to the field value.

14. An apparatus as in claim 1 in which space is reserved in JPU memory at the head of the first tuple produced by the PSDP for recording tuple length and null vector, so that the length and null vectors from the end of the tuple may be relocated to this space.

15. An apparatus as in claim 1 in which at least one PSDP is implemented as a Field Programmable Gate Array (FPGA).

16. An apparatus as in claim 1 in which the host computers in the first group contain software comprising a plan optimizer component that determines which filtering functions should be executed within a PSDP.

17. An apparatus as in claim 1 in which the JPUs in the second group contain software comprising a plan optimizer component that determines which filtering functions should be executed within a PSDP.

18. An apparatus as in claim 1 in which the host computers in the first group contain software comprising a plan link component, which determines a query execution plan, the query execution plan further having portions that will be processed by a PSDP, portions that will be processed by a JPU after a PSDP has returned data to the JPU, and portions that will be processed by a host, after the JPU has returned data to the host group.

19. An apparatus as in claim 1 in which the JPUs in the second group contain software comprising a plan link component, which determines a query execution plan, the query execution plan further having portions that will be processed by a PSDP, portions that will be processed by a JPU after a PSDP has returned data to the JPU, and portions that will be processed by a host, after the JPU has returned data to the host group.

20. An apparatus as in claim 1 in which the hosts in the first group contain software comprising a PSDPPrep component, which, for a given query execution plan, defines filtering instructions.

21. An apparatus as in claim 1 in which the JPUs in the second group contain software comprising a PSDPPrep component, which, for a given query execution plan, defines filtering instructions.

22. An apparatus as in claim 21 wherein the instructions defined by the PSDPPrep component include instructions to process fields of records.

23. An apparatus as in claim 21 in which a PSDPPrep component further identifies at least one of filtering, transformation, projection and aggregation operations to be performed by a PSDP.

24. An apparatus as in claim 21 in which a PSDPPrep component further modifies the query execution plan to specify restrict operations that are to be performed by a PSDP instead of a JPU.

25. An apparatus as in claim 1 in which the JPUs contain software comprising a PSDP Filter component, which loads an executable code image into a PSDP.

26. An apparatus as in claim 1 in which the JPUs contain software comprising a PSDP Scheduler component, which schedules jobs to run on a PSDP and queues PSDP requests to retrieve required data.

27. An apparatus as in claim 1 in which the JPUs in the second group contain software comprising a JPU Resource Scheduler component, which is responsible for scheduling jobs to be run on the JPU.

28. An apparatus as in claim 27 in which the JPU Resource Scheduler component further schedules jobs to run on a PSDP, communicating with a PSDP Scheduler component to queue up PSDP requests to retrieve required data.

29. An apparatus as in claim 27 in which the JPU Resource Scheduler component further schedules jobs, in which similar PSDP instructions in different query execution plans are combined to avoid duplicate PSDP processing requests.

30. An apparatus as in claim 2 in which an initial query is provided by a structured query language (SQL) statement, and the records specified thereby exist in various processing states within at least two components of the system, the two components including at least one of a PSDP within a JPU and a host.

31. An apparatus as in claim 30 in which a PSDP processes fields within records are received from the streaming data source, without waiting to process any records until all records are received.

EVIDENCE APPENDIX

None.

RELATED PROCEEDINGS APPENDIX

None.